BAB 2
LANDASAN TEORI

2.1 Pengenalan Penyakit Asma

Asma merupakan salah satu penyakit kronik dengan serangan *intermiten*. Serangan ditandai dengan adanya *spasme* dari saluran *bronkial*, pembengkakan dinding *bronkial* dan banyaknya *sekresilender*. Semua keadaan tersebut mengakibatkan timbulnya batuk, bunyi ngik, sesak napas dan rasa kontriksi pada dada. Tanpa pengelolaan yang baik asma akan mengganggu kehidupan penderita dan akan cenderung mengalami peningkatan, sehingga dapat menimbulkan komplikasi ataupun kematian. Walaupun asma merupakan penyakit yang dikenal luas di masyarakat namun kurang dipahami semestinya hingga timbul anggapan dari sebagian dokter dan masyarakat bahwa asma merupakan penyakit yang sederhana serta mudah diobati. Sehingga timbul kebiasaan untuk mengatasi gejala asma hanya terhadap gejala sesak napas dan mengi dengan pemakaian obat-obatan dan bukannya mengelola asma secara lengkap. Pengelolaan secara dini lebih baik untuk mengetahui kondisi kesehatan khususnya asma, adapun data yang menyatakan bahwa kondisi kesehatan tubuh terjangkit penyakit asma atau tidak berdasarkan umur bisa dilihat pada gambar 2.1.

Age (years)	Score = 0 (normal)	Score = 1 (mildly abnormal)	Score = 2 (moderately abnormal)	Score = 3 (markedly abnormal)
Respiratory rate				
<1	<35	35–44	45–54	≥55
1–3	<25	25–34	35–44	≥45
3–6	<20	20–24	25–34	≥35
Heart rate				
<1	<140	140–159	160–179	≥180
1–3	<120	120–134	135–159	≥160
3–6	<100	100–109	110–129	≥130

Gambar 2. 1 Data Potensi Penyakit Asma

2.1.1 Faktor Penyebab Asma

Asma, merupakan suatu bentuk peradangan kronis yang terjadi pada saluran pernapasan. Biasanya asma memiliki gejala-gejala yang bervariasi pada

setiap orang, yang dipicu oleh banyak faktor. Asma biasanya membuat penderitanya menjadi sesak napas dan sulit untuk bernapas karena terjadinya peradangan pada saluran pernapasan tersebut. Ada banyak sekali faktor yang bisa menjadi penyebab asma ini timbul. Berikut ini faktor penyebab asma paling utama.

1. Faktor Genetik

Salah satu faktor yang paling sering menjadi penyebab dari kambuhnya asma, atau terjangkitnya seseorang dengan penyakit asma adalah faktor genetik atau faktor bawaan. Faktor genetik atau bawaan ini diturunkan oleh generasi sebelumnya, seperti orang tua, nenek kakek, ataupun buyut. Seseorang yang mendapatkan bakat asma karena keturunan atau faktor genetik ini biasanya mengalami gejala – gejala asma yang mirip dengan orang tua atau kakek neneknya dahulu. Asma yang diperoleh karena faktor genetik atau bawaan ini lebih sulit untuk dihilangkan, hanya dapat dikurangi saja gejala – gejalanya secara bertahap.

2. Faktor Kondisi Saluran Nafas

Seperti sudah diketahui sebelumnya, asma terjadi ketika saluran pernapasan seseorang mengalami iritasi, sehingga menyempit dan menyebabkan kesulitan untuk bernapas, atau biasa disebut sesak napas. Ada beberapa orang yang memiliki kondisi saluran pernapasan yang sangat sensitif, sehingga mudah terjadi iritasi pada saluran pernapasannya. Hal ini menyebabkan mereka akan lebih mudah terserang asma, karena kondisi saluran nafasnya yang mudah teriritasi.

3. Faktor Lingkungan

Faktor lain yang sangat mempengaruhi kambuhnya penyakit asma ini adalah faktor yang berasal dari lingkungan sekitar. Lingkungan sekitar memiliki banyak jenis polutan dan hal hal lain yang sanggup membuat saluran pernapasan anda teriritasi dan mengalami penyempitan, hingga akhirnya mengalami sesak nafas.

4. Faktor Kondisi Medis

Faktor lain yang dapat menimbulkan gejala asma bagi anda adalah faktor yang berasal dari kondisi medis.Kondisi medis dapat berupa efek dari penggunaan obat – obatan tertentu, ataupun kondisi pasien yang mengalami suatu penyakit yang dapat mempengaruhi kondisi saluran pernapasan mereka.

5. Stress

Stress sudah terbukti dapat berpengaruh secara negatif dengan kondisi kesehatan seseorang. Orang- orang dengan tingkat stress yang tinggi, bahkan merasa depresi akan mengalami beberapa gangguan – gangguan kesehatan yang, salah satunya adalah mengalami penyebab asma dan sesak nafas ini.

6. Olahraga yang berlebihan

Olahraga memang sangat baik bagi kesehatan tubuh, namun olahraga yang berlebihan sangat tidak disarankan. Terutama bagi orang yang memiliki bakat sebagai penderita asma. Olahraga berlebihan akan sangat mengganggu kemampuan pernapasan, sehingga gejala asma dan sesak napas akan timbul dan sangat mengganggu aktivitas sehari-hari.

2.1.2 *Morbiditas* Penyakit Asma

Berdasarkan data WHO pada tahun 2011 mengatakan Peningkatan *prevalensi* asma di seluruh dunia secara umum dan khususnya peningkatan frekuensi perawatan pasien di RS atau kunjungan ke emergensi. Penyebab diduga disebabkan peningkatan kontak dan interaksi alergen di rumah (asap, merokok pasif) dan atmosfir (debu kendaraan). Kondisi *sosio ekonomis* yang rendah menyulitkan pemberian tempat yang baik. *Prevalensi* asma di seluruh dunia adalah sebesar 8-10% pada anak dan 3-5% pada dewasa, dan dalam 10 tahun terakhir ini meningkat sebesar 50%. *Prevalensi* asma di Jepang dilaporkan meningkat 3 kali dibanding tahun 1960 yaitu dari 1,2% menjadi 3,14%, lebih banyak pada usia muda. Penelitian *prevalensi* asma di Australia 1982-1992 yang didasarkan kepada data atopi, bunyi ngik dan HRB menunjukkan kenaikan *prevalensi* asma akut di

daerah lembab dari 4,4%(1982) menjadi 11,9% (1992). Singapura dari 3,9% (1976) menjadi 13,7% (1987), di Manila 14,2% menjadi 22.7% (1987). Data dari daerah perifer adalah sebesar 0,5% dari 215 anak dengan bakat atopi sebesar 20,5%, bunyi ngik 2%, HRB 4% (Woolcock, 1994). Serangan asma juga semakin berat, terlihat dari meningkatnya angka kejadian asma rawat inap dan angka kematian. Asma juga merubah kualitas hidup penderita dan menjadi sebab peningkatan absen anak sekolah dan kehilangan jam kerja. Biaya asma sebesar 7.000 Milyar di Perancis yaitu 1% dari biaya pemeliharaan kesehatan langsung ataupun tidak langsung meningkat terus (Michel, 1995). Menurut Soena Stxmantri (1995) penelitian di Indonesia sering menggunakan kuesioner dan jarang dengan pemeriksaan HRB. Hampir semuanya dilakukan di lingkungan khusus misalnya di sekolah atau rumah sakit dan jarang di lingkungan masyarakat. Dilaporkan pasien asma dewasa di RS Hasan Sadikin berobat jalan tahun 1985-1989 sebanyak 12.1% dari jumlah 1.344 pasien dan 1993 sebanyak 14,2% dari 2.137 pasien. Pada perawatan inap 4,3% pada 1984-1985 dan 7,5% pada 1986-1989. Pasien asma anak dan dewasa di Indonesia diperkirakan sekitar 38%, Survei Kesehatan Rumah Tangga (SKRT) 1986 mengajukan angka sebesar 7,6%. Hasil penelitian asma pada anak sekolah berkisar antara 6,4% dari 4.865 dan 15,15% dari 1.515 anak.

2.1.3 Sistem Pengelolaan Asma

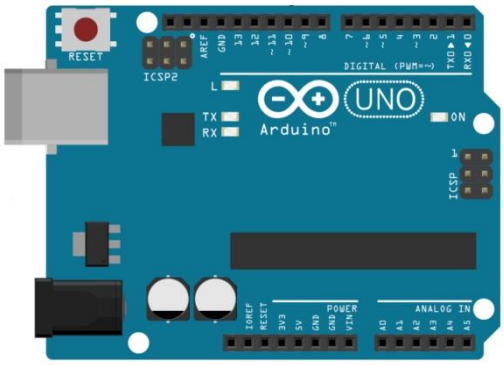
Di dalam buku "*International consensus Report on Diagnosis and management of Astma*. *US Departement of Health and Human Services. Clinical and Experimental Allergy, Vol. 2. Suppl. I.*" Pengelolaan asma belum menyeluruh terhadap berbagai aspeknya secara sistimatik dan kontinyu. Terapi belum tuntas dan umumnya baru ditujukan untuk mengatasi gejala asmanya saja. Pengelolaan secara sistimatis seharusnya mencakup:

1. Penegakan diagnosis lengkap, tingkat beratnya asma.
2. Kerjasama yang kontinyu antara dokter (klinik/RS) dengan pasien dan lingkungannya (di rumah dan tempat kerja).
3. Upaya mengatasi serangan dan terapi pencegahan di klinik/RS dan di rumah.

4. Pilihan obat yang tepat berupa suatu sistim dengan pemilihan *steroid* sebagai terapi asma utama yang ditujukan untuk mengatasi *inflamasi* pada semua tingkat asma, kecuali yang paling ringan.
5. Membiasakan tindak lanjut dengan pengukuran dari *peak flow meter*, melaksanakanusaha rehabilitasi.
6. Upaya pengelolaan asma yang dilakukan secara gigih dan teratur.

2.2 Pengenalan *Microcontroller Arduino*

Arduino adalah pengendali mikro *single-board* yang bersifat *open-source*, diturunkan dari *Wiring platform*, dirancang untuk memudahkan penggunaan elektronik dalam berbagai bidang. Perangkat kerasnya memiliki prosesor Atmel AVR dan perangkat lunaknya memiliki bahasa pemrograman C. *Data sheet arduino* bisa dilihat pada gambar 2.2.



Gambar 2. 2 Rangkaian *Arduino*

Arduino digunakan pada penelitian ini adalah sebagai penerima data informasi detak jantung melalui *pulse sensor* dan data informasi frekuensi nafas melalui sensor nafas yang di ambil dari penderita yang kemudian dikirim ke *android* melalui *bluetooth*.

Rangkaian yang ada di board *arduino* diantaranya yaitu :

1. 14 *pin input/Output digital* (0-13)

   Berfungsi sebagai *input* atau *Output*, dapat diatur oleh program. Khusus untuk 6 buah *pin* 3, 5, 6, 9, 10 dan 11, dapat juga berfungsi sebagai *pin*

analog output dimana tegangan *Output*-nya dapat diatur. Nilai sebuah *pin Output analog* dapat diprogram antara 0 – 255, dimana hal itu mewakili nilai tegangan 0 – 5V.

2. *USB (Universal Serial Bus)*

Berfungsi untuk: Memuat program dari komputer ke dalam papan dan Komunikasi *serial* antara papan dan komputer.

3. Sambungan SV1

Sambungan atau *jumper* untuk memilih sumber daya papan, apakah dari sumber *eksternal* atau menggunakan *USB.* Sambungan ini tidak diperlukan lagi pada papan *Arduino* versi terakhir karena pemilihan sumber daya *eksternal* atau *USB* dilakukan secara otomatis.

4. Q1 – Kristal (*quartz crystal oscillator*)

Jika *microcontroller* dianggap sebagai sebuah otak, maka *kristal* adalah jantung-nya karena komponen ini menghasilkan detak-detak yang dikirim kepada *microcontroller* agar melakukan sebuah operasi untuk setiap detak-nya. *Kristal* ini dipilih yang berdetak 16 juta kali per detik (16MHz).

5. *In-Circuit Serial Programming (ICSP)*

Port ICSP memungkinkan pengguna untuk memprogram *microcontroller* secara langsung, tanpa melalui *bootloader*. Umumnya pengguna *Arduino* tidak melakukan ini sehingga *ICSP* tidak terlalu dipakai walaupun disediakan.

6. IC 1 – *Microcontroller Atmega*

Komponen utama dari papan *Arduino,* di dalamnya terdapat *CPU, ROM* dan *RAM*. Jika hendak disuplai dengan sumber daya *eksternal*, papan *Arduino* dapat diberikan tegangan *DC* antara 9-12V.

7. 6 *pin input analog* (0-5)

Pin ini sangat berguna untuk membaca tegangan yang dihasilkan oleh sensor *analog*, seperti sensor suhu. Program dapat membaca nilai sebuah *pin input* antara 0 – 1023, dimana hal itu mewakili nilai tegangan 0 – 5V. *Pin analog* yang digunakan pada penelitian ini yaitu *pin* A0 digunakan untuk membaca nilai tegangan yang dihasilkan oleh sensor pulsa sebagai

pembaca detak jantung, dan *pin* A1 digunakan untuk membaca *Output* dari sensor suara (*microphone*).

2.2.1 Pemrograman *Arduino*

Arduino pada umumnya bekerja menggunakan pemrograman dengan bahasa C yang dituliskan pada *software Arduino IDE*. *SoftwareIDE arduino* terdiri dari 3 bagian:

1. *Editor program*, yaitu tempat untuk penulisan atau pengeditan program yang akan di tanamkan pada *Arduino*. Setiap program *Arduino* biasa disebut *sketch*.
2. *Compiler*, yaitu modul yang berfungsi mengubah bahasa pemrograman kedalam kode *biner*, karena hanya kode *biner* yang dapat dipahami *microcontroller*.
3. *Uploader*, yaitu modul yang berfungsi memasukan kode *biner* kedalam memori mikrokontroller.

Setiap program *Arduino* (biasa disebut dengan *sketch*) mempunyai dua buah fungsi yang harus ada , yaitu:

1. *Void setup*( ) { }

   Semua kode yang berada di dalam kurung kurawal akan dijalankan hanya satu kali ketika program *Arduino* dijalankan untuk pertama kalinya.
2. *Void loop*( ){ }

   Fungsi ini akan dijalankan setelah *setup* (fungsi *void setup*) selesai. Setelah dijalankan satu kali fungsi ini akan dijalankan lagi, dan lagi secara terus menerus sampai catu daya dilepaskan.

2.3 Pengenalan Sensor Pulsa (*Pulse Sensor*)

Sensor pulsa adalah sebuah sensor denyut jantung yang dirancang untuk *Arduino*. Sensor ini dapat mendeteksi denyut nadi pada telapak tangan dengan cara menggabungkan data denyut jantung ke dalam aplikasi yang telah dibuat. Tegangan keluaran *Pulse sensor* adalah 3-5 *volt* dan pada saat arus 4 ma membutuhkan 5 *volt*. Alat ini menggunakan *filter* dan *Op-Amp* untuk meningkatkan

amplitudo dari pulsa gelombang dan menormalisasi sinyal ke titik referensi. Ketika sensor tidak dalam kontak dengan sumber denyut jantung keluaran dari sinyal tersebut berada di titik tengah dari tengangan atau V/2. Ketika sensor menyentuh sumber denyut nadi maka akan berubah menjadi cahaya yang dipantulkan ketika darah di pompa melalui jaringan dan akan membuat sinyal berfluktuasi di sekitar titik referensi. Sensor Pulsa dirancang untuk mengukur IBI *(Inter Beat Interval).* IBI adalah selang waktu pada denyut jantung dalam mili detik dengan waktu momen sesaat dari jantung berdetak. BPM *(Beat per Minute)* berasal setiap detak dari rata-rata setiap 10 kali IBI. Jadi, ketika *microcontroller arduino* dinyalakan dan berjalan dengan sensor pulsa yang dicolokkan ke *pin analog* 0, terus-menerus (setiap 2 mS) membaca nilai sensor berdasarkan denyut jantung yang terukur. Sehingga nilai *BPM* dirumuskan :

$$BPM = \frac{60000}{\left(\frac{rata - rata\ nilaiIBI}{10}\right)}$$

Gambar 2. 3 Rumus *Beat Per Minute* (Gari D. Clifford)

Pada peneltian ini, rumus diatas digunakan pada saat pengkodingan *hardware arduino* untuk mengolah data yang didapat dari sensor pulsa yang kemudian data diinterpretasikan sebagai frekuensi detak jantung per menit (*Heart beat rate*). Ketika sensor pulsa ditempelkan pada ujung jari, setiap dua mili detik sekali sensor pulsa akan mengeluarkan nilai tegangan yang kemudian akan dibaca oleh *arduino* melalui *pin analog* A0. Data yang dihasilkan berupa deret angka berpola naik turun, ketika pola naik (*amplitudo*) itu menunjukan terjadinya detak jantung, waktu yang ditempuh diantara dua *amplitudo* itu yang disebut dengan *IBI (Inter Beat Interval).* *IBI* akan dirata-ratakan setelah terdeteksi 10 kali *amplitudo*, kemudian nilai dari rata-rata *IBI* dijadikan pembagi untuk satu menit yang akan menghasilkan nilai *BPM* (*Beat per Minute*).

2.4 Pengenalan Sensor Suara

Sensor suara adalah sebuah alat yang mampu mengubah gelombang Sinusioda suara menjadi gelombang sinus energi listrik (*Alternating Sinusioda Electric Current*). Alat pendeteksi sinyal suara bekerja berdasarkan prinsip pemfilteran suara yang didengar oleh komponen mikrofon. Sinyal analog hasil pembacaan mikrofon akan disaring dengan menggunakan *unit band pass filter* yang meloloskan sinyal *analog*. Sensor suara bekerja berdasarkan besar/kecilnya kekuatan gelombang suara yang mengenai membran sensor yang menyebabkan bergeraknya membran sensor yang juga terdapat sebuah kumparan kecil di balik membran tadi naik dan turun. Oleh karena kumparan tersebut sebenarnya adalah ibarat sebuah pisau berlubang-lubang, maka pada saat ia bergerak naik-turun, ia juga telah membuat gelombng magnet yang mengalir melewatinya terpotong-potong. Kecepatan gerak kumparan menentukan kuat-lemahnya gelombang listrik yang dihasilkannya. Dalam penelitian ini sensor suara digunakan untuk merekam data frekuensi pernafasan (*Respiratory rate*).

2.5 Pengenalan *Bluetooth*

Bluetooth adalah standar teknologi nirkabel untuk pertukaran data jarak pendek. *Bluetooth* menggunakan teknologi radio yang disebut *frekuency-hopping spread spectrum*. Data yang ditransmisikan dibagi menjadi paket-paket dan setiap paket ditransmisikan pada salah satu dari 79 saluran *bluetooth* yang ditunjuk. Pada penelitian ini *bluetooth* digunakan sebagai media transmisi pengiriman data dari *arduino* ke *android*. Bisa dilihat pada gambar 2.4.



Gambar 2. 4 *Bluetooth*

2.6 Pengenalan *Android*

Android Android adalah sistem operasi yang berbasis Linux untuk telepon seluler seperti telepon *pin*tar (*Smartphone*) dan komputer tabelt. *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti begerak. Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan *Android*, dibentuklah *Open Handset Alliance*, *konsorsium* dari 34 perusahaan piranti keras, piranti lunak dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile dan Nvidia. Bebrapa pengertian lain dari *Android*, yaitu :

1. Merupakan *platform* terbuka (*Open source*) bagi para *programmer* untuk membuat atau mengembangkan aplikasi.
2. Merupakan sistem operasi yang dibeli Google Inc dari *Android* Inc.
3. Bukan bahasa pemrograman, akan tetapi hanya menyediakan lingkungan hidup atau *runtime environment* yang disebut DVM (*Dalvik Virtual Machine*) yang telah dioptimasi untuk *device* dengan sistem memori yang kecil.

2.6.1 Versi *Android*

Banyak *smartphone* dan PC Tabelt menggunakan sistem operasi dengan versi yang berbeda. Semakin tinggi versi, fiturnya semakin canggih dan banyak. Telepon pertama yang memakai sistem operasi *Android* adalah HTC Dream 13 yang dirilis pada tanggal 22 Oktober 2008. Beberapa uraian versi *android* seperti dibawah ini :

1. ***Android* 1.1**

   OS *android* pertama versi 1.0 di rilis pada tanggal 23 bulan September tahun 2008 dan *android* versi 1.1 Dirilis pada tanggal 9 bulan Februari tahun 2009 di namakan dengan *Android*.

2. ***Android* versi 1.5**

   OS *android* versi 1.5 Dirilis pertama pada tanggal 30 bulan April tahun 2009 di namakan dengan *Android Cupcake*.

3. ***Android* versi 1.6**

   OS *android* versi 1.6 Dirilis pada tanggal 15 bulan September tahun 2009 di namakan dengan *Android Donut*.

4. ***Android* versi 2.0**

   OS *android* versi 2.0 Dirilis pada tanggal 26 bulan Oktober tahun 2009 dan beberapa versi lainya seperti *android* v2.0.1 dan *android* V2.1 di namakan dengan *Android Eclair*.

5. ***Android* versi 2.2**

   OS *android* versi 2.2 dan versi v2.2.3 Dirilis pada tanggal 10 bulan mei tahun 2010 di namakan dengan *Android Froyo*.

6. ***Android* versi 2.3**

   OS *android* versi 2.3 sampai dengan versi v2.3.7 Dirilis pertama pada tanggal 6 bulan Desember tahun 2010 di namakan dengan *Android Gingerbread*. *android Gingerbread* menjadi salah satu os *android* tersukses di tahun 2010, hingga saat ini masih banyak sekali ponsel *android* yang menggunakan versi ini.

7. ***Android* versi v3.0**

   OS *android* versi v3.0 sampai dengan versi v3.2 Dirilis pertama pada tanggal 22 bulan Februari tahun 2011 di namakan dengan *Android Honeycomb*, os *android Honeycomb* merupakan os *android* paling sempurna di bandingkan dengan os *android* versi sebelumnya, os ini terkenal akan kinerja os yang sangat baik dan memiliki tampilan mewah. Jadi tidak heran apabila di tahun 2011 ponsel yang menggunakan *android Honeycomb* yang paling banyak di cari.

8. ***Android* versi v4.0**

   OS *android* versi v4.0 sampai dengan v4.0.2 Dirilis pertama pada tanggal 19 bulan Oktober tahun 2011 di namakan dengan *Android Ice Cream Sandwich*. ini merupakan versi Os *Android* tersukses dan paling banyak cari, tampilan pada *Android Ice Cream Sandwich* sudah tidak terlalu jauh dengan *android jelly bean* tersebut, dan sampai saat ini *android Ice Cream Sandwich* masih banyak di gunakan pada ponsel kelas menengah kebawah dan beberapa

samsung galaxy tab terdahulu. kelebihan utama dari *android ice crean sandwich* ini adalah sudah mendukung *Flash player* sehingga bisa memudahkan kita browsing di internat dll.

9. ***Android* versi v4.1**

   OS *android* versi v4.1 sampai dengan v4.3 Dirilis pertama pada tanggal 9 bulan juli tahun 2012 di namakan dengan *Android Jelly Bean*. *Android Jelly Bean* merupakan salah satu os yang di perbaharui dari versi *ice cream sandwich*, dari segi tampilan os *android jelly bean* paling sempurna di banding beberapa os terdahulu, tidak hanya memiliki tampilan yang lebih dinamis, *android jelly bean* juga telah di optimasi pada bagian kinerja os super cepat. Di tambah dengan beberapa peningkatan kinerja pada sisi kamera pada *smartphone* dan *tabelt pc*, penambahan *navigasi Gesture* dan banyak lagi, sampai di tahun 2014 ini *android jelly bean* masih menjadi salah satu os yang paling banyak di gunakan dan masih banyak sekali ponsel terbaru *android* yang masih menggunakan os *android jelly bean* satu ini.

10. ***Android* versi v4.4**

    OS *android* versi v4.4 sampai dengan v4.4.4 Dirilis pertama pada tanggal 31 bulan Oktober tahun 2013 di namakan dengan *Android kitkat*. Os *android kitkat* memiliki tampilan 100% lebih dinamis dan berbeda total dengan *android jelly bean*, *android* kitkat di optimasi pada sisi konsumsi baterai dan kinerja os lebih cepat ketika di jalankan pada perangkat memiliki spesifikasi lebih rendah, seperti kita tahu jika *andorid jelly bean* memiliki kelebihan pada sisi konsumsi baterai yang lebih tinggi dan ketika di jalankan di perangkat yang memiliki versi rendah os ini tidak maksimal.

11. ***Android* versi 5.0**

    OS *android* versi v5.0 dirilis pertama pada tahun 2014 dikenal dengan nama "*Android Lollipop*" *android lollipop* memiliki peningkatan sistem keamanan dan tampilan lebih dinamis, os versi terbaru dari *android* ini sepertinya bisa di gunakan berbagai perangkat elektronik seperti tv dll, untuk mengetahui beberapa kelebihan os *android lollipop* versi v5.0 silahkan baca disini.

12. ***Android* versi 6.0**

Android versi 6.0 dikenal dengan *Marshmallow* dikenal juga dengan nama *android* M, os ini resmi di rilis pada 28 Mei tahun 2015, dari segi tampilan hampir mirip dengan *android* versi 5.0 *lollipop*, hanya saja sudah di fitur keamanan di os ini sudah di tingkatkan. Untuk lebih *detail* tentang os *android* terbaru ini baca di kelebihan dari os *Android* versi 6.0 *Marshmallow*.

Pada penelitian ini perangkat telepon *pin*tar *android* dibutuhkan untuk menjalankan aplikasi pendeteksi dini penyakit asma. *Android* berkomunikasi dengan alat *ardunio* melalui komunikasi *bluetooth*. Adapun spesifikasi perangkat yang dibutuhkan adalah sebagai berikut:

Tabel 2. 1 Spesifikasi Perangkat *Android*

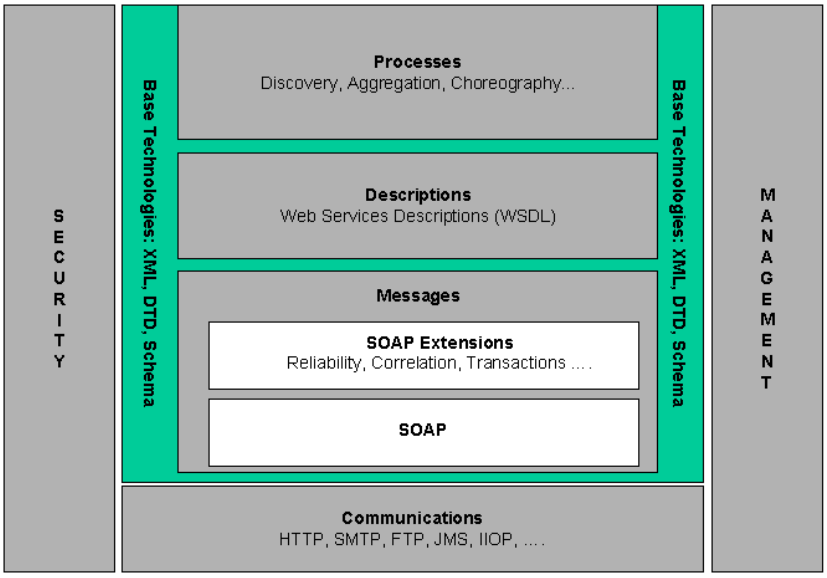
Processor	*Dual-core 1.0 GHz Cortex-A7*
RAM	*1GB*
ROM	*2GB*
Storage	*microSD 1GB*
Network	*GSM / HSPA*
Bluetooth	*V4.0, A2DP*
Screen	*Capacitive Touchscreen*
Operating Sistem	*4.2 Jelly Bean*

2.7 Pengenalan *Web Service*

Web service adalah sebuah entitas komputasi yang dapat diakses melalui jaringan internet maupun intranet dengan standar protokol tertentu dalam *platform* dan antarmuka bahasa pemrograman yang independen. Tujuan pengembangannya adalah untuk menjembatani komunitas antar program, sehingga aplikasi yang satu dan aplikasi yang lain yang terdapat pada suatu jaringan yang sama atau pada

jaringan berbeda dapat saling berkomunikasi asalkan menggunakan standar protokol yang ditetapkan oleh *webservice*. Hal ini bisa terjadi, karena standar protokol itu tidaklah terikat pada suatu *platform* atau bahasa pemograman. Protokol itu sendiri dibangun oleh *Extensible Markup Language* (XML) yang memang kenyataannya telah didukung oleh banyak *paltform*, bahasa pemograman, dan oleh *developer* di seluruh dunia.

Arsitektur *web service* dibangun oleh beberapa layer dan teknologi yang saling berhubungan. Banyak cara untuk memvisualisaskan *service*, sama banyaknya dengan cara untuk membangun dan menggunakan *web service* gambar berikut hanyalah salah satu cara untuk menggambarkan arsitekturnya. Bisa dilihat pada gambar 2.5.



Gambar 2. 5 Arsitektur *Web service*

Web service disusun oleh tiga komponen standart, yaitu :

1. *Simple Object Access Protocol* (SOAP), yaitu protokol yang bertanggungjawab dalam pertukuran informasi dalam lingkungan jaringan terdistribusi.

2. *Web service definition Language* (WDSL), dokumen standar yang dituliskan pada format XML, dan mendifinikasan kehadiran *web service* dalam suatu jaringan.
3. *Universal description, Discovery and Inttegration* (UDDI), yaitu suatu lokasi direktori yang beriskan *service* (layanan) dan bersifat bebas platform (*platform independent*), dituliskan berbasis XML dan dapat diakses oleh entittas yang berada di dalam dan luar jaringan.

Adanya *standart* tersebut membuat *web service* mudah di akses melalui berbagai antarmuka dan juga memberi peluang dimungkinkannya berbagai sistem yang dibangun pada *platform* yang berbeda dan bahasa yang berbeda untuk berkolaborasi pada dalam suatu pekerjaan.

2.8 Pengenalan JSON

JSON (*Java Script Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman *Java Script*, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh *programmer* keluarga C termasuk *C, C++, C#, Java, Java Script, Perl, Python* dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data.

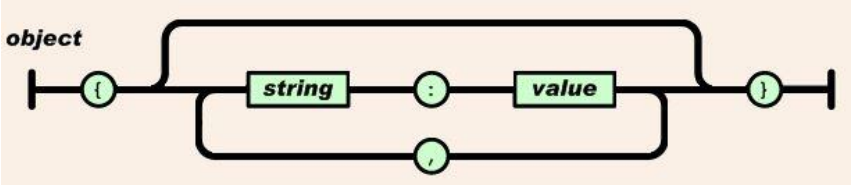
JSON terbuat dari dua struktur yaitu:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel *hash* (*hash tabel*), daftar berkunci (*keyed list*), atau *associative array*.
2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

Struktur-struktur data ini disebut sebagai struktur data *universal*. Pada dasarnya, semua bahasa pemprograman *modern* mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini.

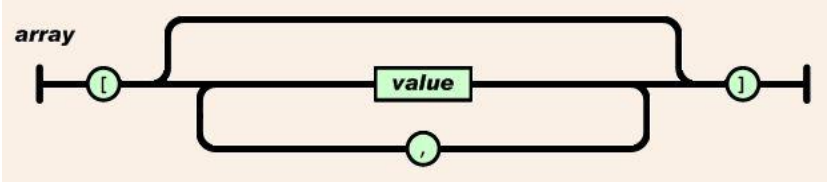
JSON menggunakan bentuk sebagai berikut:

1. **Objek** adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma). Bisa dilihat pada gambar 2.6.



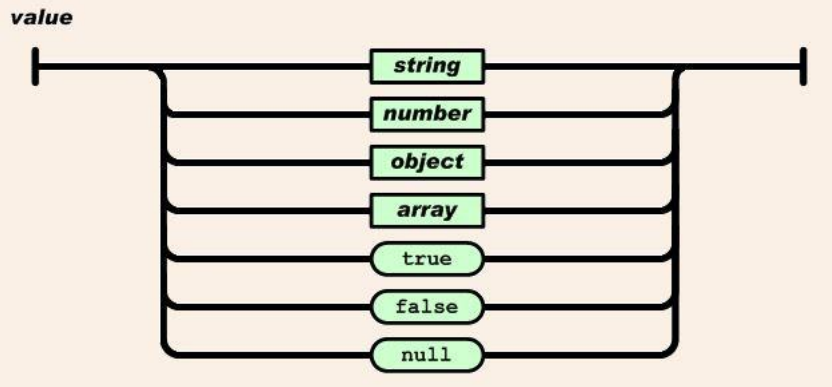
Gambar 2. 6 Bentuk Objek JSON

2. ***Array*** adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma). Bisa dilihat pada gambar 2.7.



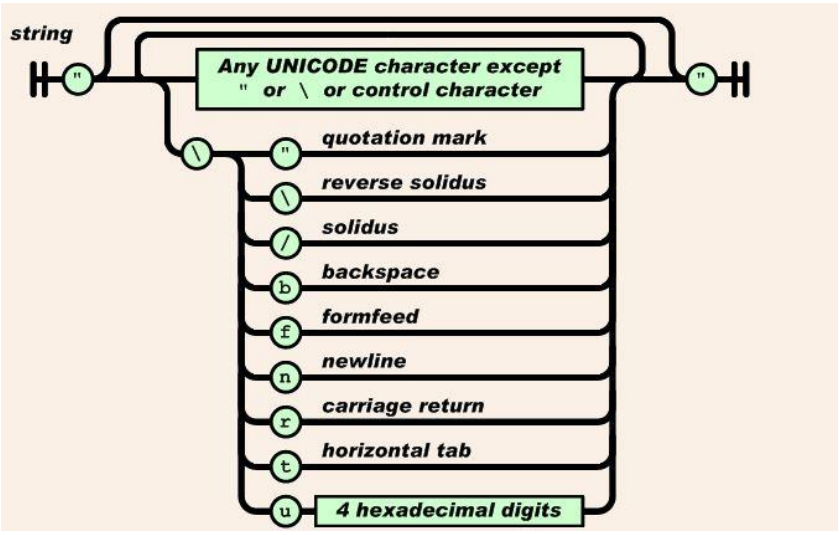
Gambar 2. 7 Bentuk *Array* JSON

3. **Nilai (*value*)** dapat berupa sebuah *string* dalam tanda kutip ganda, atau angka, atau *true* atau *false* atau *null*, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat. Bisa dilihat pada gambar 2.8.



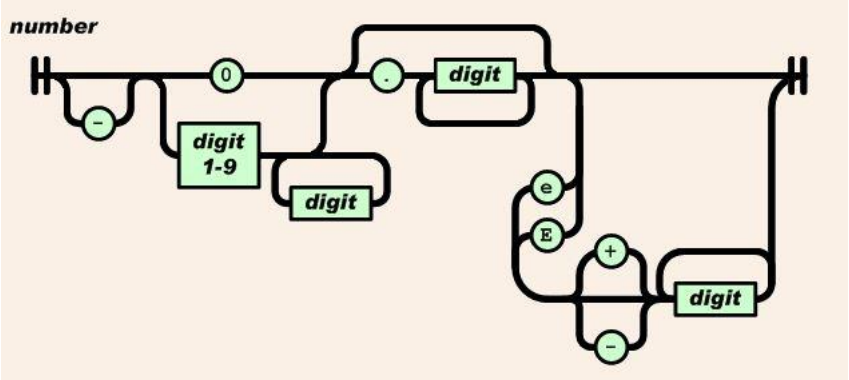
Gambar 2. 8 Bentuk *Value* JSON

4. ***String*** adalah kumpulan dari nol atau lebih karakter *Unicode*, yang dibungkus dengan tanda kutip ganda. Di dalam *string* dapat digunakan *backslash escapes* "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada *string*. *String* sangat mirip dengan string *C* atau *Java*. Bisa dilihat pada gambar 2.9.



Gambar 2. 9 Bentuk *String* JSON

5. **Angka** adalah sangat mirip dengan angka di *C* atau *Java*, kecuali format oktal dan heksadesimal tidak digunakan. Bisa dilihat pada gambar 2.10.



Gambar 2. 10 Bentuk Angka JSON

Spasi kosong (*whitespace*) dapat disisipkan di antara pasangan tanda-tanda tersebut, kecuali beberapa detil *encoding* yang secara lengkap dipaparkan oleh bahasa pemprograman yang bersangkutan.

2.9 *Prototype*

Prototype adalah proses pembuatan model sederhana *software* yang mengijinkan pengguna memiliki gambaran dasar tentang program serta melakukan pengujian awal.

2.10 *Prototyping*

Metode *prototyping* adalah sistem informasi yang menggambarkan hal-hal penting dari sistem informasi yang akan datang. *Prototype* sistem informasi bukanlah merupakan sesuatu yang lengkap, tetapi sesuatu yang harus dimodifikasi kembali, dikembangkan, ditambahkan atau digabungkan dengan sistem informasi yang lain.

Tahapan *Prototyping* :

1. Pengumpulan kebutuhan

   Pada penelitian ini pengumpulan kebutuhan dilakukan untuk mengumpulkan data-data yang mendukung pengembangan aplikasi pendeteksi penyakit asma. Meliputi pengumpulan data parameter frekuensi detak jantung dan frekuensi pernafasan pada penderita penyakit asma dan orang normal.

2. Membangun *prototyping*

   Membangun *prototyping* dengan membuat perancangan sementara yang berfokus pada perekaman detak jantung dan pernafasan, menentukan sensor yang digunakan serta perancangan sementara tampilan aplikasi pada layar telepon *pin*tar *android*.

3. Evaluasi *protoptyping*

   Mengevaluasi *prototype* yang telah dirancang apakah sesuai dengan kebutuhan aplikasi yang akan dikembangkan yaitu pendeteksi penyakit asma. Jika sudah sesuai maka langkah ke empat akan diambil jika tidak *prototype* direvisi dengan mengulang langkah 1,2,3.

4. Mengkodekan sistem

   Pengkodingan dilakukan setelah perancangan *prototype* telah benar-benar disepakati. Pada penelitian ini terdapat dua tahap pengkodingan yaitu pengkodingan pada *arduino* (mengolah data sensor) dan pada *android* (Penyajian Data berbasis HTML5).

5. Menguji sistem

   Setelah perangkat lunak dibangun, maka dilakukan pengujian untuk menguji tingkat kehandalan perangkat lunak yang telah dibangun. Hal ini dilakukan untuk memastikan kehandalan *software*. Pada penelitian ini dilakukan pengujian dengan metode *black box testing*.

6. Evaluasi Sistem

   Evaluasi dilakukan lagi setelah *prototyping* diimplementasikan. Jika sudah sesuai maka langkah ke tujuh akan diambil jika tidak mengulang langkah 4,5,6.

7. Menggunakan sistem

   Aplikasi telah melalui tahap uji dan siap digunakan masyarakat umum sebagai aplikasi pendeteksi dini dan mandiri penyakit asma.

Keunggulan *Prototyping* :

1. *User* dapat berpartisipasi aktif.
2. Penentuan kebutuhan lebih mudah diwujudkan.
3. Mempersingkat waktu pengembangan SI.

Kelemahan *Prototyping* **:**

1. Proses analisis dan perancangan terlalu singkat.
2. Mengesam*ping*kan alternatif pemecahan masalah.
3. Bisanya kurang fleksibel dalam mengahadapi perubahan.
4. *Prototype* yang dihasilkan tidak selamanya mudah dirubah.
5. *Prototype* terlalu cepat selesai.

2.11 Pengenalan UML Sebagai Pemodelan Sistem

Unified Modeling Language (UML) digunakan untuk melakukan pemodelan sistem/perangkat lunak dengan menggunakan *tools* yang ada. Dengan pemodelan menggunakan UML, rekayasa dan pengembangan perangkat dapat dilakukan dengan fokus pengembangan dan desain perangkat lunak terhadap :

1. Tinjauan umum bagaimana arsitektur sistem secara keseluruhan.
2. Penelaah bagaimana objek-objek dalam sistem saling mengirimkan pesan (*message*) dan saling bekerjasama satu sama lain.
3. Menguji apakah sistem/perangkat lunak sudah berfungsi seperti yang seharusnya.
4. Dokumentasi sistem/perangkat lunak untuk keperluan-keperluan tertentu di masa yang akan datang.

Setiap sistem yang komplek seharusnya bisa dipandang dari sudut pandang yang berbeda-beda sehingga bisa dilakukan pemahaman secara menyeluruh. Dalam upaya-nya tersebut, UML menyediakan sembilan jenis diagram yang dapat dikelompokkan berdasarkan sifatnya yang statis ataupun dinamis. Kesembilan jenis diagram untuk UML adalah:

1. ***Use-Case Diagram* -** bersifat statis, memperlihatkan himpunan *use-case* dan aktor-aktor dihubungkan oleh *relationship* yang digambarkan garis. Diagram ini sangat penting terutama untuk memodelkan ataupun mengorganisasikan perilaku dari sistem yang dibutuhkan pengguna.

   a. Use Case

   Use Case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *Use Case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu, bisa dilihat pada gambar 2.11.



Gambar 2. 11 Bentuk *Use Case*

b. Aktor

Aktor adalah sesuatu (*entitas*) yang berhubungan dengan sistem dan berpartisipasi dalam *use case*. Aktor menggambarkan orang, sistem atau entitas *Eksternal* yang secara khusus membangkitkan sistem dengan *input* atau masukan kejadian-kejadian, atau menerima sesuatu dari sistem. Bisa dilihat pada gambar 2.12.

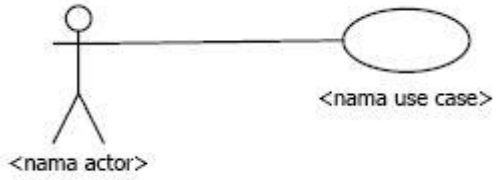


Gambar 2. 12 Bentuk Aktor

c. Relationship

Relasi (*relationship*) digambarkan sebagai bentuk garis antara dua simbol dalam *use case* diagram. Relasi antara *actor* dan *use case* disebut juga dengan asosiasi (*association*). Asosiasi ini digunakan untuk menggambarkan

bagaimana hubungan antara keduanya. Relasi-relasi yang terjadi pada *use case* diagram bisa antara *actor* dengan *use case* atau *use case* dengan *use case*. Bisa dilihat pada gambar 2.13.



Gambar 2. 13 Bentuk *Reationship*

Relasi antara *use case* dengan *use case* :

a. ***Include***, pemanggilan *use case* oleh *use case* lain atau untuk menggambarkan suatu *use case* termasuk di dalam *use case* lain (diharuskan). Contohnya adalah pemanggilan sebuah fungsi program. Digambarkan dengan garis lurus berpanah dengan tulisan <<*include*>>.
b. ***Extend***, digunakan ketika hendak menggambarkan variasi pada kondisi perilaku normal dan menggunakan lebih banyak kontrol *form* dan mendeklarasikan *ekstension* pada *use case* utama. Atau dengan kata lain adalah perluasan dari *use case* lain jika syarat atau kondisi terpenuhi. Digambarkan dengan garis berpanah dengan tulisan <<*extend*>>.
c. ***Generalization/Inheritance***, dibuat ketika ada sebuah kejadian yang lain sendiri atau perlakuan khusus dan merupakan pola berhubungan *base-parent use case*. Digambarkan dengan garis berpanah tertutup dari *base use case* ke *parent use case*.

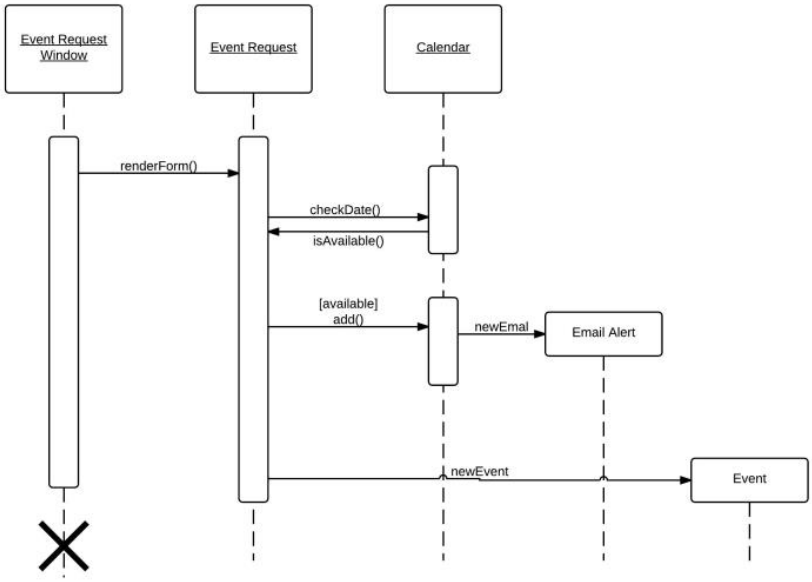
2. **Model *use case* (*use case scenario*)**

Model ini menggambarkan bagaimana system bekerja secara keseluruhan dan pada skenario tertentu. Sehingga jika sebuah *object* merupakan sesuatu yang menyediakan layanan maka harus dapat dibuktikan bahwa semua layanan tersebut ada dalam suatu *use case*. Model ini digambarkan dengan diagram *use case scenario*.

3. ***Activity Diagram*** **-** bersifat dinamis. Merupakan tipe khusus dari *diagram state* yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem

4. ***Class Diagram*** **-** bersifat statis tetapi sering pula memuat kelas-kelas aktif dan memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi.

5. ***Sequence Diagram*** **-** bersifat dinamis yang menekankan pada pengiriman pesan (*message*) dalam suatu waktu tertentu. Contoh *Sequence* Diagram bisa dilihat pada gambar 2.16.



Gambar 2. 14 Contoh *Sequence* Diagram

2.12 Perangkat Lunak Pendukung

Adapun beberapa perangkat lunak pendukung dalam penyusunan laporan Pembangunan aplikasi pendeteksi dini penyakit asma dengan *Android* ini adalah :

2.12.1 Intel XDK

Intel XDK merupakan salah satu *software developer* kit dari Intel yang digunakan untuk mengembangkan aplikasi HTML5 *hybrid* kedalam berbagai perangkat *mobile* dan aplikasi HTML5 lainnya, seperti *ekstensi* google chrome, aplikasi facebook, atau sebuah *website mobile*. Pada *software* Intel XDK terdapat perangkat untuk melakukan *develop, emulate, debbuging, testing* dan *build* aplikasi yang telah dikembangkan menjadi aplikasi *web* dan aplikasi HTML5 *hybrid* untuk berbagai *platform mobile* diantaranya iOS, *Android,* Tizen, Windows 8 Store, Windows Phone 8, Amazon, Nook. Berikut beberapa tampilan dari Intel XDK :

Develop, adalah tahapan dari penulisan (*coding*) *script* terhadap suatu aplikasi yang akan dibuat.



Gambar 2. 15 Tampilan Develope Intel XDK

Emulate, adalah tahapan dimana kita melakukan sebuah *preview* terhadap aplikasi yang telah selesai penulisan (*coding*) scriptnya.

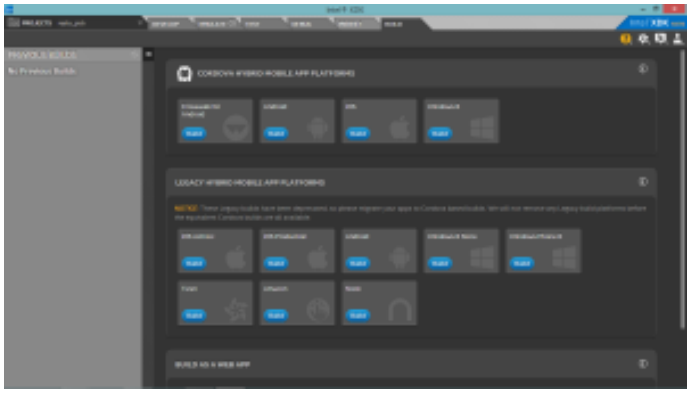
Test, adalah suatu tahapan dimana kita melakukan ujicoba secara langsung terhadap aplikasi yang telah dibuat ke dalam perangkat *mobile* yang telah dihubungkan ke komputer/laptop menggunakan kabel data.

Debug, adalah suatu tahapan yang dilakukan untuk mencari dan mengurangi *bug*, atau mencari kerusakan dari sebuah aplikasi atau perangkat keras melalui perangkat *mobile* yang telah dihubungkan ke komputer/laptop menggunakan kabel data. Sehingga sebuah aplikasi dapat berjalan sesuai dengan yang diharapkan.



Gambar 2. 16 *Emulator* Intel XDK

Build, adalah tahapan terakhir dari Intel XDK. Yaitu melakukan kompilasi (*compile*) program yang telah dibuat agar dapat digunakan sesuai perangkat *mobile* yang diinginkan.



Gambar 2. 17 Tampilan Build Intel XDK

2.12.2 HTML5

HTML5 adalah sebuah bahasa markah untuk menstrukturkan dan menampilkan isi dari *World Wide Web*, sebuah teknologi inti dari Internet. HTML5 adalah revisi kelima dari HTML (yang pertama kali diciptakan pada tahun 1990 dan versi keempatnya, HTML4, pada tahun 1997) dan hingga bulan Juni 2011 masih dalam pengembangan. Tujuan utama pengembangan HTML5 adalah untuk memperbaiki teknologi HTML agar mendukung teknologi multimedia terbaru, mudah dibaca oleh manusia dan juga mudah dimengerti oleh mesin.

HTML5 merupakan salah satu karya *Konsortium World Wide Web Consortium* untuk mendefinisikan sebuah bahasa markah tunggal yang dapat ditulis dengan cara HTML ataupun XHTML. HTML5 merupakan jawaban atas pengembangan HTML 4.01 dan XHTML 1.1 yang selama ini berjalan terpisah, dan diimplementasikan secara berbeda-beda oleh banyak perangkat lunak pembuat *web*.

Di masa depan, format HTML5 akan menjadi sangat penting bagi penyedia layanan *browser* dan *web* desain, dalam memajukan alat perambannya menjadi lebih kaya aplikasi, *web* dan *web* desain yg interaktif dan menarik. Demikian pengakuan dari *General Manager Internet Explorer* (IE) Dean Hachamovitch. Spesifikasi HTML5 menggambarkan dukungan video tanpa menentukan format video tertentu. Kami berpikir H.264 adalah format yg sangat baik untuk mendukung HTML5 nya, IE9 nantinya juga akan mendukung pemutaran video H.264 saja.

H.264 adalah suatu standar industri, dgn dukungan *hardware* yg luas dan kuat. Karena standarisasi ini, Anda dapat dgn mudah menerima apa yg Anda rekam, meskipun dengan video pada kamera biasa, lalu meletakannya di situs dengan sistem operasi atau perangkat dengan dukungan H.264. Karena semua alasan tersebut, kita fokus HTML5 video dan dukungan pada H.264. Kendati format video akan sangat menarik jika dipasangkan dengan H.264, Hachamovitch mengakui bahwa mayoritas video internet saat ini masih didominasi berbasis *Flash*. *Flash* memang memiliki beberapa masalah, khususnya di sekitar keandalan, keamanan, dan kinerja. Meskipun saat ini, *Flash* tetap merupakan bagian penting dalam memberikan pengalaman konsumen yg terbaik di situs hingga hari ini.

2.12.2.1 Sejarah HTML5

Kelompok Kerja Teknologi Aplikasi *Web Hyperteks* (*Web Hypertext Application Technology Working Group*, WHATWG) mulai membuat standar baru ini pada tahun 2004 ketika Konsortium W3C sedang fokus pada pengembangan XHTML 2.0 di masa depan, sementara HTML 4.01 belum pernah diperbarui sejak tahun 2000. Sejak tahun 2009, W3C dan WHATWG bekerja sama dalam pengembangan HTML5 setelah W3C mengakhiri Kelompok Kerja Pengembangan XHTML 2.0. Meskipun HTML5 telah dikenal luas oleh para pengembang *web* sejak lama, HTML5 baru mencuat pada April 2010 setelah CEO Apple Inc., Steve Jobs, mengatakan bahwa dengan pengembangan HTML5, "*Adobe Flash* sudah tidak dibutuhkan lagi untuk menyaksikan video atau menyaksikan konten apapun di *web*.

2.12.2.2 Fitur HTML5

1. ***Canvas*** **-** Dulu, untuk bisa memberikan interaksi menggambar di halaman *web* kita harus memakai applet *Java* atau *Flash*. HTML 5 akan memberikan satu opsi tambahan: *canvas*. Seperti namanya, *canvas* adalah media yang bisa dicorat-coret langsung. Tidak lagi perlu memuat plugin khusus. Cukup tambahkan dan *Javascript* maka kita sudah bisa menggambar langsung di halaman *web*. Sekarang Anda bisa berimaginasi sendiri, kira-kira apa saja yang orang lakukan dengan. Apa yang sebelumnya jadi monopoli *Flash* dan *aplet Java* akan di-*take-over* oleh.

2. **Video dan Audio -** Akan ada tag dan di HTML 5. Jadi tidak perlu lagi menempelkan *Flash* untuk sekedar memutar audio. Format video yang didukung akan bervariasi terhadap *browser*, kemungkinan besar *codecnya* adalah Ogg Theora (*patent free*) dan H.264. Sepertinya sampai sekarang *codecnya* masih jadi kontroversi.

3. ***Local Storage*** **-** Sekarang *storage* untuk *browser* akan diakomodasi sebagai *standard* dalam HTML 5. Aplikasi bisa menyimpan data dalam jumlah lebih besar dari biasanya tanpa harus mengimplementasikan trik dengan *cookie* atau *Flash*. Tentunya ini kabar baik bagi pengembang aplikasi *web*. Mungkin

bisa meningkatkan performa aplikasi dengan menggunakan *storage* sebagai *local cache*.

4. ***Web Workers*** **-** Jika *Javascript* biasanya yang kita nikmati di *web* kadangkala menyebabkan komputer kita melambat atau paling tidak membuat *browser* seperti sesak napas maka *web worker* akan bsia jadi pelega. Salah satu fitur *web worker* adalah *threading*. Kini *Javascript* bisa dipakai untuk melakukan beberapa proses sekaligus tanpa harus menghambat proses terkait UI.
5. ***Semantics*** **-** Untuk para *designer* yang sering meng-*abusediv* dan *span* sebagia elemen *nav, fretnomore*. Akan ada tag khusus untuk navigasi, *section*, *footer*, dll. Tag yang kaya *semantic* seperti ini pasti akan lebih bermanfaat dari pada tag yang hanya punya informasi format dan *layout* saja. Dan bagi mesin, HTML5 akan jadi lebih bisa dimengerti.

2.12.3 Pengenalan PHP

PHP merupakan salah satu bahasa pemrograman *server-side* diantara beberapa pemrograman yang ada. Sejak diluncurkan, PHP mendapat *respon* yang sangat baik dari kalangan pengembang aplikasi *web*. Kemudahan untuk dipahami, serta sintaknya yang mirip bahasa *C* menjadikan pemrograman ini cepat dikenal oleh kalangan luas.

PHP pertama kali dibuat oleh **Rasmus Lerdorf** pada tahun 1995. Pada waktu itu PHP bernama **FI** (*Form Interpreted*). Pada saat tersebut PHP adalah sekumpulan *script* yang digunakan untuk mengolah data *form* (hasil *input User*) dari *web*.

Perkembangan selanjutnya adalah **Rasmus** melepaskan kode sumber tersebut dan menamakannya **PHP/FI**, pada saat tersebut kepanjangan dari PHP/FI adalah *Personal Home Page/Form Interpreter*. Dengan pelepasan kode sumber ini menjadi *open source*, maka banyak *programmer* yang tertarik untuk ikut mengembangkan PHP.

Pada November 1997, dirilis **PHP/FI 2.0**. Pada rilis ini *interpreter* sudah diimplementasikan dalam bahasa *C*. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan. Pada tahun 1997, sebuah perusahaan bernama ***Zend*** (asal kata dari pendirinya : ***Zeev Suraski***

dan ***Andi Gutmans***) menulis ulang *interpreter* PHP menjadi lebih bersih, lebih baik dan lebih cepat. Kemudian pada Juni 1998 perusahaan tersebut merilis *interpreter* baru untuk PHP dan meresmikan nama rilis tersebut menjadi **PHP 3.0**.

PHP 3.0 telah ditulis ulang dari awal. Dia telah memiliki peningkatan yang sepantasnya dengan kode yang lebih kuat dan konsisten dibandingkan dengan versi 2.0. 3.0 juga lebih cepat secara signifikan, dan menggunakan sedikit memori. Bagaimanapun juga, beberapa peningkatan ini tidak akan mungkin terjadi tanpa perubahan kompatibilitas, yaitu kedua *sintax* dan fungsinya.

Tambahannya, para pengembang PHP telah mencoba untuk membersihkan keduanya pada fungsi PHP yaitu *sintax* dan semantik pada versi 3.0, dan ini juga telah pula menyebabkan beberapa penyesuaian. Pada pergerakan panjang ini, dipercaya bahwa perubahan-perubahan ini untuk melangkah yang lebih baik pada masa yang akan datang.

Pada pertengahan tahun 1999, *Zend* merilis *interpreter* PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai. Versi ini banyak dipakai sebab versi ini mampu dipakai untuk membangun aplikasi *web* kompleks tetapi tetap memiliki kecepatan proses dan stabilitas yang tinggi.

PHP 4 dengan *Zend engine* yang terintegrasi telah meningkatkan performansi dan kapabilitas dari PHP secara besar, tetapi perawatan sangat besar telah dilakukan untuk menghilangkan kerusakan pada kode yang ada sekecil mungkin. Jadi bermigrasi pengkodean dari PHP 3 ke PHP 4 akan lebih mudah daripada bermigrasi dari PHP/FI 2 kepada PHP 3. Banyak sekali kode PHP 3 yang telah ada sebelumnya sudah siap dijalankan tanpa perubahan, tetapi kita tetap harus mengetahui beberapa perubahan kecil dan melakukan pemeliharaan untuk mengetes pengkodean yang kita buat sebelum mengganti versi dalam lingkungan produksi pengkodean. Perkembangan yang terjadi seharusnya memberikan kita gambaran tentang apa yang akan kita cari dan buat pada pengkodean pemrograman PHP ini.

Pada 13 Juli 2004, *Zend* merilis PHP 5.0. Versi ini adalah versi mutakhir dari PHP. Dalam versi ini, inti dari *interpreter* PHP mengalami perubahan besar. Dalam versi ini juga dikenalkan model pemrograman berorientasi objek baru untuk

menjawab perkembangan bahasa pemrograman kearah pemrograman berorientasi objek.

PHP 5 *Zend Engine* 2 yang terintegrasi telah secara besar meningkatkan dayaguna dan kemampuan PHP, tetapi pemeliharaan yang besar telah dilakukan untuk kerusakan sekecil mungkin pengkodean sebisa mungkin. Jadi bermigrasi kode kita dari PHP 4 ke PHP 5 akan sangat mudah. Kebanyakan kode PHP 4 yang ada telah seharusnya siap untuk dijalankan tanpa perubahan-perubahan, tetapi kita harus sedikit tahu tentang perubahan dan merawat dan mengetes pengkodean kita sebelum berubah versi pada lingkungan produksi kita.

Tanpa mengurangi kemampuan-kemampuan yang sudah ada, PHP 5 hadir dengan fitur yang lebih kompleks, dan merupakan rilis terbaru yang dikeluarkan oleh pengembangnya. Secara besar-besaran pengembangan PHP merombak *bug* yang sering dijumpai dalam PHP 4, dan hal ini akan sangat terasa ketika kita membuat aplikasi berbasis objek.

2.12.4 Pengenalan MySQL

Database MySQL merupakan sistem basis data yang sangat terkenal dan bersifat *Open Source* artinya bahwa semua orang diizinkan menggunakan dan memodifikasi *software* tersebut. MySQL dibangun, didistribusikan, dan didukung oleh MySQL AB. Sebenarnya *software* MySQL mempunyai dua macam lisensi, yaitu yang pertama bersifat *Open Source* dengan menggunakan GNU (*General Public Licence*) dan lisensi kedua itu berupa lisensi komersial standar (*standard commercial licence*) yang dapat dibeli dari MySQL AB.

MySQL merupakan sistem manajemen *database* atau basis data terhubung (*relational database management system*). *Database* terhubung menyimpan data pada tabel – tabel terpisah. Hal tersebut akan menambah kecepatan dan fleksibilitasnya. Kata SQL pada MySQL merupakan singkatan dari "*Structured Query Language*". SQL merupakan bahasa standar yang digunakan untuk mengakses *database* dan ditetapkan oleh ANSI / ISO SQL *Standard*.

Fitur yang ada pada MySQL adalah sebagai berikut :

1. Ditulis dalam bahasa C dan C++.
2. Bekerja dalam berbagai *platform* (misalnya Mac OS X, Solaris, Sun OS, UNIX, Novel Netware, Windows, dan lain-lain).
3. Menyediakan mesin penyimpan (*engine storage*) transaksi dan nontransaksi,
4. *Server* tersedia sebagai *platform* program yang terpisah untuk digunakan pada lingkungan jaringan *clien/server*.
5. MySQL mempunyai *library* yang dapat ditempal pada aplikasi yang berdiri sendiri (*standalone application*) sehingga aplikasi tersebut dapat digunakan pada komputer yang tidak mempunyai jaringan.
6. Mempunyai sistem *password* yang fleksibel dan aman.
7. Dapat menangani basis data dalam skala besar. Besar data dalam *server* MySQL dapat berisi 50 juta *record*.
8. Klien dapat terkoneksi ke MySQL *Server* menggunakan *socket TCP/IP* pada *platform* manapun,
9. *Software* dapat mengirim pesan kesalahan ke klien dalam berbagai bahasa.

MySQL mempunyai logo berupa ikan lumba – lumba yang bernama "*Sakila*". Nama tersebut merupakan kontribusi dan pemenang kontes "*nama lumba – lumba*" yang diadakan oleh MySQL AB. "*Sakila*" berasal dari bahasa lokal *Swaziland*, "*Sakila*" juga merupakan nama kota di *Arusha Tanzania*.

2.13 Metode Pengujian

Metode pengujian yang dilakukan dalam penelitian ini menggunakan beberapa metode diantaranya menggunakan metode *black box testing*, *white box testing*, dan pengujian sampel. Pengujian dilakukan untuk mengetahui sistem apakah sudah berjalan dengan baik dengan fungsi – fungsi sesuai perancangan, dan juga untuk mengidentifikasi kesalahan-kesalahan yang ditemukan pada saat eksekusi program.

2.13.1 Pengujian *Black box*

Menurut Black (2009 :3), *Tester* menggunakan *behavioral test* (disebut juga *Black-Box Tests*), sering digunakan untuk menemukan *bug* dalam *high level operations*, pada tingkatan fitur, profil operasional dan skenario *customer*. *Tester* dapat membuat pengujian fungsional *black box* berdasarkan pada apa yang harus sistem lakukan. *Behavioral testing* melibatkan pemahaman rinci mengenai domain aplikasi, masalah bisnis yang dipecahkan oleh sistem dan misi yang dilakukan sistem. *Behavioral test* paling baik dilakukan oleh penguji yang memahami desain sistem, setidaknya pada tingkat yang tinggi sehingga mereka dapat secara efektif menemukan *bug* umum untuk jenis desain. Menurut Nidhra dan Dondeti (2012:1), *black box testing* juga disebut *functional testing*, sebuah teknik pengujian fungsional yang merancang test case berdasarkan informasi dari spesifikasi.

2.13.2 Pengujian *White box*

White box testing adalah pengujian yang didasarkan pada pengecekan terhadap *detail* perancangan, menggunakan struktur kontrol dari desain program secara prosedural untuk membagi pengujian ke dalam beberapa kasus pengujian. Secara sekilas dapat diambil kesimpulan *white box testing* merupakan petunjuk untuk mendapatkan program yang benar secara 100%.

2.13.3 Pengujian Sampel

Pada penelitian ini dilakukan tahapan pengujian sistem pada sampel, Sampel adalah bagian dari jumlah dan karakteristik yang dimiliki oleh populasi tersebut (Sugiyono, 2011). Dalam pengambilan sampel ini penulis menggunakan teknik *sampling purposive*, yaitu teknik pengambilan sampel dengan pertimbangan tertentu sesuai dengan kebutuhan penelitian (Sugiyono, 2011). Berdasarkan pengertian tersebut penulis menentukan karakteristik sampel yang dibutuhkan dalam penelitian ini yaitu sampel dengan usia 1 sampai 6 tahun yang memiliki riwayat medis normal (tidak memiliki penyakit asma), dan sampel dengan usia 1 sampai 6 tahun yang memiliki riwayat penyakit asma. Kemudian sistem akan diujikan ke masing-masing sampel yang telah ditentukan.